B. AMENDMENTS TO THE CLAIMS

In order to better assist the Examiner with the prosecution of the case, the current pending claims have been included in their entirety for which reconsideration is requested.

1. (Currently Amended) A method for efficient packet desegmentation on a network adapter, comprising:

enabling a network adapter at a computer system to control a flow of a plurality of data packet segments over a bus system to a network protocol stack running at the computer system in each of individual packet segments and desegmented groups of packet segments, wherein each of said plurality of data packet segments is separately received by said network adaper from a network over a plurality of separate connections to a plurality of separate other computer systems, wherein each of said plurality of separate connections is identified by a separate selection of addresses and ports;

buffering a first data packet segment from a single connection from among said plurality of separate connections in the network adapter;

responsive to detecting at least one next data packet segment from said single connection, buffering [[a]] the at least one next ~~plurality of~~ data packet segment[[s]] received at said [[a]] network adapter from said single connection, wherein said single connection is identified by a matching plurality of addresses and ports extracted from each header of each of said first data packet segment and said at least one next [[plurality of]] data packet segment[[s]]; and

responsive to detecting a buffering release condition, releasing said selection of said plurality of data packet segments from said network adapter in a single traversal over said bus system flagged as a desegmented group to said [[a]] network protocol stack for processing flagged desegmented groups of data packet segments together, such that data packet[[s]] segments received from said single connection are efficiently passed to said network stack to be processed together by said network protocol stack.

2. (Cancelled).

3. (Original) The method according to claim 1 for efficient packet desegmentation wherein said single connection is a TCP connection identified by a four-tuple of source and destination addresses and ports extracted from each TCP header of each of said plurality of data packet segments.

4. (Original) The method according to claim 1 for efficient packet desegmentation further comprising:

detecting said buffering release condition when a new data packet segment received at said network adapter is from a different connection than said single connection.

5. (Original) The method according to claim 1 for efficient packet desegmentation further comprising:

detecting said buffering release condition when a time a first receiving data packet segment from among said plurality of data packet segments is buffered at said network adapter exceeds a time threshold.

6. (Original) The method according to claim 1 for efficient packet desegmentation further comprising:

detecting said buffering release condition when a queue size limit in said network adapter for buffering data packet segments is reached.

7. (Original) The method according to claim 1 for efficient packet desegmentation further comprising:

detecting said buffering release condition when an abnormal condition occurs, wherein said abnormal condition is at least one from among a checksum mismatch, a connection reset, an urgent pointer, and a missing packet being detected.

8. (Currently Amended) A system for efficient packet desegmentation on a network adapter, comprising:

a network adapter with an interface for controlling a flow of a plurality of data packet segments over a bus system to a network protocol stack running at the computer system in each of individual packet segments and desegmented groups of packet segments, wherein each of said plurality of data packet segments is separately received by said network adaper from a network over a plurality of separate connections to a plurality of separate other computer systems, wherein each of said plurality of separate connections is identified by a separate selection of addresses and ports ~~facilitating transfer of data packets between a data processing system and a network~~;

said network adapter further comprising:

a buffer for buffering a first data packet segment from a single connection from among said plurality of separate connections in the network adapter;

said [[a]] buffer, responsive to detecting at least one next data packet segment from said single connection, for buffering [[a]] the at least one next ~~plurality of~~ data packet segment[[s]] received at said [[a]] network adapter from said single connection across said network, wherein said single connection is identified by a matching plurality of addresses and ports extracted from each header of each of said first data packet segment and said at least one next [[plurality of]] data packet segment[[s]]; and

a desegmenting means for releasing said selection of said plurality of data packet segments in a single traversal over said bus system from said buffer together flagged in a desegmented group to [[a]] said network protocol stack in said data processing system for processing flagged desegmented groups of data packet segments together, responsive to detecting a buffering release condition.

9. (Cancelled).

10. (Original) The system according to claim 8 for efficient packet desegmentation wherein said single connection is a TCP connection identified by a four-tuple of source and destination addresses and ports extracted from each TCP header of each of said plurality of data packet segments.

11. (Original) The system according to claim 8 for efficient packet desegmentation, said desegmenting means further comprising:

means for detecting said buffering release condition when a new data packet segment received at said network adapter is from a different connection than said single connection.

12. (Original) The system according to claim 8 for efficient packet desegmentation, said desegmenting means further comprising:

means for detecting said buffering release condition when a time a first receiving data packet segment from among said plurality of data packet segments remains within said buffer exceeds a time threshold.

13. (Original) The system according to claim 8 for efficient packet desegmentation, said desegmenting means further comprising:

means for detecting said buffering release condition when a queue size limit in said buffer is reached.

14. (Original) The system according to claim 8 for efficient packet desegmentation, said desegmenting means further comprising:

means for detecting said buffering release condition when an abnormal condition occurs, wherein said abnormal condition is at least one from among a

checksum mismatch, a connection reset, an urgent pointer, and a missing packet being detected.

15. (Currently Amended) A computer program product for efficient packet desegmentation on a network adapter, comprising:

a volatile or non-volatile recording medium;

means, recorded on said recording medium, for enabling a network adapter at a computer system to control a flow of a plurality of data packet segments over a bus system to a network protocol stack running at the computer system in each of individual packet segments and desegmented groups of packet segments, wherein each of said plurality of data packet segments is separately received by said network adaper from a network over a plurality of separate connections to a plurality of separate other computer systems, wherein each of said plurality of separate connections is identified by a separate selection of addresses and ports;

means, recorded on said recording medium, for buffering a first data packet segment from a single connection from among said plurality of separate connections in the network adapter;

means, recorded on said recording medium, responsive to detecting at least one next data packet segment from said single connection, for buffering [[a]] the at least one next ~~plurality of~~ data packet segment[[s]] received at said [[a]] network adapter from said single connection, wherein said single connection is identified by a matching plurality of addresses and ports extracted from each header of each of said first data packet segment and said at least one next [[plurality of]] data packet segment[[s]]; and

means, recorded on said recording medium, for releasing said selection of said plurality of data packet segments from said network adapter in a single traversal over said bus system flagged as a desegmented group to said [[a]] network protocol stack for processing flagged desegmented groups of data packet segments together, responsive to detecting a buffering release condition.

16. (Canceled).

17. (Original) The computer program product according to claim 15 for efficient packet desegmentation further comprising:

means, recorded on said recording medium, for detecting said buffering release condition when a new data packet segment received at said network adapter is from a different connection than said single connection.

18. (Original) The computer program product according to claim 15 for efficient packet desegmentation further comprising:

means, recorded on said recording medium, for detecting said buffering release condition when a time a first receiving data packet segment from among said plurality of data packet segments is buffered at said network adapter exceeds a time threshold.

19. (Original) The computer program product according to claim 15 for efficient packet desegmentation further comprising:

means, recorded on said recording medium, for detecting said buffering release condition when a queue size limit in said network adapter for buffering data packet segments is reached.

20. (Original) The computer program product according to claim 15 for efficient packet desegmentation further comprising:

means, recorded on said recording medium, for detecting said buffering release condition when an abnormal condition occurs, wherein said abnormal condition is at least one from among a checksum mismatch, a connection reset, an urgent pointer, and a missing packet being detected.

21. (Newly Added) The method according to claim 1, further comprising:

responsive to receiving said flagged data packet segments together at said network protocol stack, processing said flagged data packet segments together by only executing an incoming packet processing code executed for each incoming data packet segment once for said flagged data packet segments, by only performing a direct memory access performed for transferring each packet segment to a memory block once for said flagged data packet segments, and by only performing a protocol control block search to detect a connection status for the connection of each data packet segment once for detecting a status of said single connection of said flagged data packet segments.

22. (Newly Added) The system according to claim 8, further comprising:

said network protocol stack, responsive to receiving said flagged data packet segments together at said network protocol stack, for processing said flagged data packet segments together by only executing an incoming packet processing code executed for each incoming data packet segment once for said flagged data packet segments, by only performing a direct memory access performed for transferring each packet segment to a memory block once for said flagged data packet segments, and by only performing a protocol control block search to detect a connection status for the connection of each data packet segment once for detecting a status of said single connection of said flagged data packet segments.

23. (Newly Added) The computer program product according to claim 15, further comprising:

means, recorded on said recording medium, responsive to receiving said flagged data packet segments together at said network protocol stack, for processing said flagged data packet segments together by only executing an incoming packet processing code executed for each incoming data packet segment once for said flagged data packet segments, by only performing a direct memory access performed for transferring each packet segment to a memory block once for said flagged data packet segments, and by only performing a protocol control block search to detect a connection status for the connection of each data packet segment once for detecting a status of said single connection of said flagged data packet segments.